Effectively Placing Block Replicas of Big Data On The Rack By Implementing Block Racking Algorithm

Thirunavukarasu B, Sangeetha K, Kalaikumaran T, Karthik S

***Abstract* — In current scenario Organization are supposed to work with huge amount of data. Based on those data analysis, predictions, manipulations are made. Replicas of blocks are created to improve the Redundancy in distributed system. Block Racking Algorithm was implemented to make the replicas to get placed on the racks that contain a set of Datanodes in an effective way**

***Index Terms*— Big data, Blocks Racking Algorithm, Replicas, Blocks, HDFS Architecture, MapReduce, File Systems, Hadoop, Datanodes, NameNodes, Task Tracker, Job tracker.**

IINTRODUCTION

A. Big Data

Big data is an unstructured large set of data even more than peta byte data. Unstructured data is a data which is in the form of logs. There won't be any items like row, column, etc., Big Data can be of only digital one. Data Analysis become more complicated because of their increased amount of data set. Certain tools are used in order to gain business analysis on their data. Predictions, analysis, requirements etc., are the main things that should be done using the unstructured big data. Big data is a combination of three v's those are namely Volume, Velocity and Variety. Big data will be basically processed by the powerful computer. But due to some scalable properties of the computer, the processing gets limited.

B. Hadoop

Hadoop is a framework of tools. The main objective of Hadoop is to support running of applications on big data. Hadoop is an open source tool distributed under Apache licence. It should have the efficiency to work with big volume of data (Volume), data coming in high speed (Velocity) and data of all sort of variety. Hadoop breaks the large set of data into pieces. It breaks the computation as well into smaller pieces. Once all these computations are finished their results are combined together. Hadoop have two main components, they are

- MapReduce
- File system

***Thirunavukarasu B**, ComputerScience and Engineering Department, SNS College of Technology.Coimbatore, INDIA.*

***Sangeetha K, Assistant Profressor**,Computer Science and Engineering Department, SNS College of Technology, Coimbatore, INDIA.*

***Dr Kalaikumaran T, Professor and Head,** Computer Science and Engineering Department, SNS College of Technology, Coimbatore, INDIA*

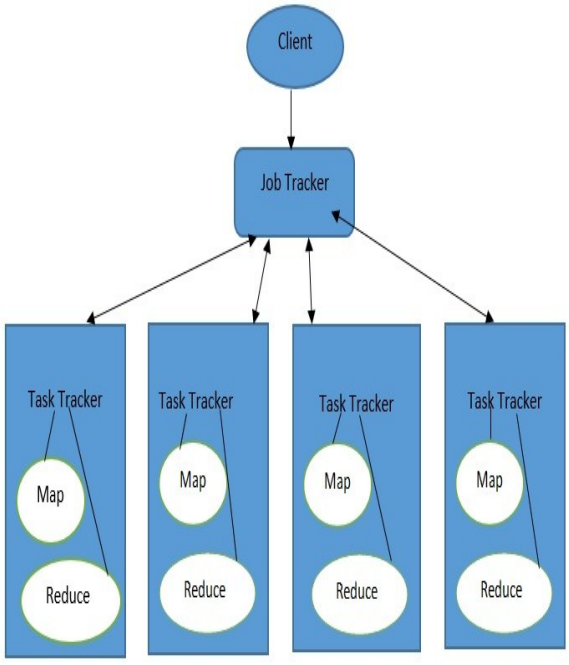
***Dr Karthik, Professor and Dean,** Computer Science and Engineering Department, SNS College of Technology, Coimbatore, INDIA.*

Where file system is called HDFS. Many tools or projects are grouped and gathered under Hadoop and the objective of this projects is to perform the task. Hadoop works on thedistributed model. And basically Hadoop is an LINUX based set of tools and hence we have LINUX on these low cost numerous computers. And all these computers will be called as slaves.

C. MapReduce

MapReduce is the heart of Hadoop. It is basically a programming model that can be written in language of choice. Mostly it is practiced to use java, python, etc. Map in the case process the data locally on the DataNode. Key value pair is generated by the Map that helps to process the data.

This key value pair reduces the work and generates the output data. Basically the file is divided into input splits. Each input split must be given to unique DataNodes. For this each input splits one needs Map. And hence number of maps is equal to number of splits. The map performs the operations required to be done with the data whereas Reduce will reduces the number of map program created. The reducing will increase the efficiency. The input splits is equal to number of Maps but the reduced size was not the same. The reduction is based on certain algorithms that make the wright reduction. There are plenty of MapReduce Algorithms, among which Google MapReduce Algorithm is most widely followed. Minimal MapReduce, sorting, searching, BFS, TF-IDF are some of the other used MapReduce Algorithms.



1. Fig. 1 Map Reduce inside the Task Tracker

D. HDFS

HDFS is a file system designed for storing very large files with streaming data access patterns, running clusters on commodity hardware. Where HDFS is not a goal fit may be,

- Low latency data access
- Lots of small files
- Multiple writers arbitrary file modifications

We cannot store the data fetched using Hadoop, Hadoop is used only for reading out huge data set in single shot.

E. HDFS Components:

- Namenode
- Datanode

Namenode

Namenode associated with the job tracker, Job tracking operations are done on the masternode or Namenode. Namenode is a reliable machine which is used to maintain and manage the blocks which present on the Datanodes. It does not store any data. The Namenode is an expensive one and have double and triple redundant machines.

Datanodes

Datanodes are the slaves of the Hadoop distributed file system. They are deployed and each machine provide octal storage. Responsible for serving read and write requests from the client. Datanode associated with the task tracker, Task tracking operations are done on the data node or commodity hardware. Task Tracker sends a heartbeat to task tracker to indicate that they are still alive.

II EXISTING METHODOLOGY:

A. HDFS Architecture:

In the existing HDFS Architecture, we mainly focussing on the blocks and Racks. The racks are the place where a group of DataNodes are arranged. These racks are of multiple in order to make use of replication. Every time the operation is performed, DataNode gets the redundant data due to replication. So certain storage is maintained to keep track of already fetched data.

HDFS mainly concentrates on the reliable storage of very large amount of data across nodes in a larger cluster. It stores the data as the sequence of blocks. It should be noted that all the blocks in a file except the last one should be of same size. These blocks of the files given by the client are replicated or fault tolerance. HDFS is designed in such a manner that it chooses the replica that is very closer to the reader or the user. Replica of local data centre is mostly preferred then choosing the remote data centre. Initially the namenode gets placed in the safe mode. During the safe mode, namenode does not create any Replication of data blocks. The task tracker sends the job tracker of Namenode a Heartbeat and this makes the Namenode to know that the corresponding Data node is free and a job can be given to the node. The main aim of HDFS architecture is to store the data perfectly even in the conditions of various failures. The three main failures that occurs on the system are

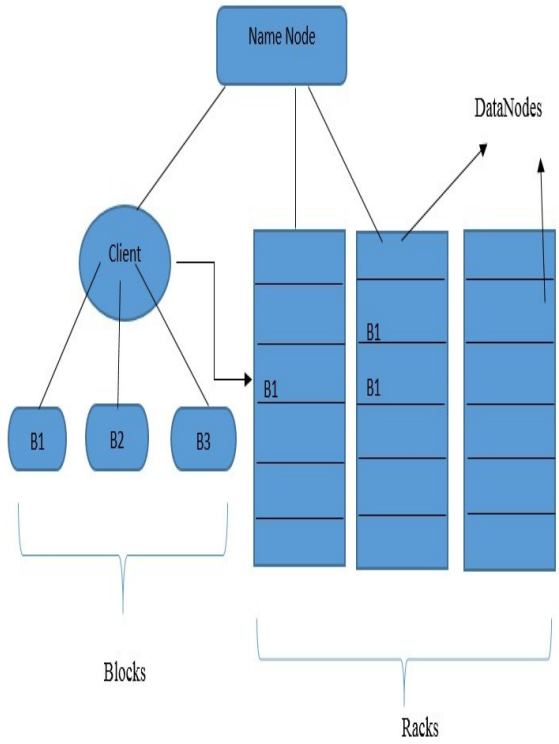
- NameNode Failure
- DataNode Failure
- Network Partitions

Due to some misconfigure of IP one make lose the access to NameNode. Namenode failure occurs when the Main server or MasterNode stops to do its works by any cause. NameNode have the Secondary NameNode that keeps the important older copy of the failed NameNode. By the worst case this Secondary NameNode can be used for recovery.

B. Communication

HDFS get placed on the top layer of TCP/IP protocol. The data transaction among the various nodes and users takes place through this TCP/IP protocol. The connection is made using the TCP protocol which responds with the acknowledge message. The communication between the DataNode and namenode is a must one. By this communication alone, the namenode can came to know that the DataNode is set free only with the help of these kind of communication among them. The protocol used for this communication is DataNode protocol.

C. Rack awareness



2. Fig. 2 Existing Replica allocation on Racks

Namenode determines the DataNode to write the 1st block to. If the client is running on a dataNode, it will try to place it there. Otherwise it chooses random order to place it within the same rack. By default, data is replicated to two other places in the cluster. A pipeline is built between the three Datanodes that makeup the pipeline. The second DataNode is randomly chosen node on the rack other than that of the first replica of the block. This is to increase redundancy. The third replica is placed in the random node within the same rack as the second replica placed. The data is piped from the second DataNode to the third one.

To ensure that the write was successful before continuing, acknowledgement packets are sent back from the third

DataNode to the second one, from the second dataNode to the first. And from the first dataNode to the client. Thus acknowledgement reaches the client indicating that the blocks are received successfully. This process occurs for the each blocks that make up the file, in this case, the second and third block. It should be noted that, for every block, there is a replica on at least two racks. Now when the client is done writing the dataNode pipeline and has received acknowledgements, it tells the NameNode that it is complete. The NameNode will check that the blocks are at least minimally replicated before responding.

In this methodology if the first DataNode gets dead due to some reasons then, only the replica of the second rack can be used. This increases fetching time as well. The client needs to move to other racks located elsewhere to fetch the DataNode. There in that rack one could find two replicas. The client can use any replica. Once this was done, the performed operation and data should be sent to client from the second rack. This needs higher amount of bandwidth for effective performance and the data transfer. This can be considered as one of the drawback in HDFS architecture.

IIIPROPOSED METHODOLOGY

In my proposed method the above mentioned drawback can be resolved. I propose with this basic example considering three blocks and three Datanodes. For a single block, three replicas are made. Number of replicas is equal to the number of Datanodes present in the system. The client gives the instruction that needs to be operated. These instructions are gathered by the Namenode. And the Namenode decides which Datanode should be selected and acknowledges the client with those informations. Based on the acknowledgement given by the Namenode, the operations are performed on the Datanodes.

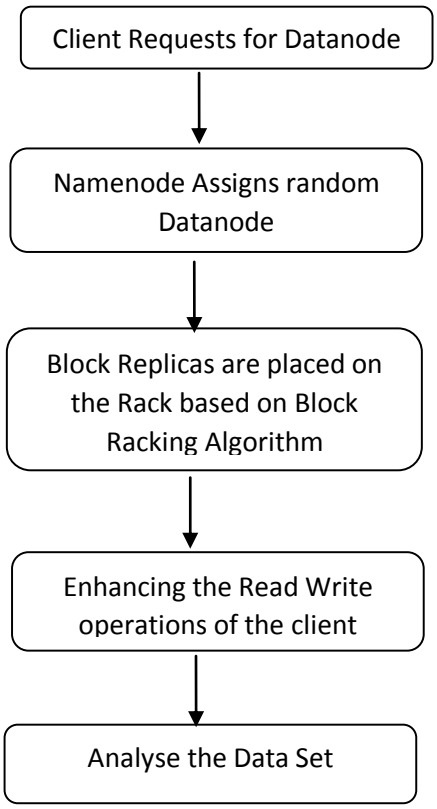


Fig. 3 Flow of Proposed Methodology

DataNode Failures that are mainly caused by the local disk failure. This DataNode failure can be overcome by making the Replicas. Network partition are mainly caused when there exists a break in communication made using TCP protocol. The protocol makes the effective flow of data. We should increase the bandwidth between the DataNode and the Client. By the same time we no need to provide such a big bandwidth between the DataNode and NameNode.

Unlike placing single replica of the block on the first rack, two replica are placed on any two Datanodes of the first rack. This Datanode on the rack is selected randomly for replica placing.

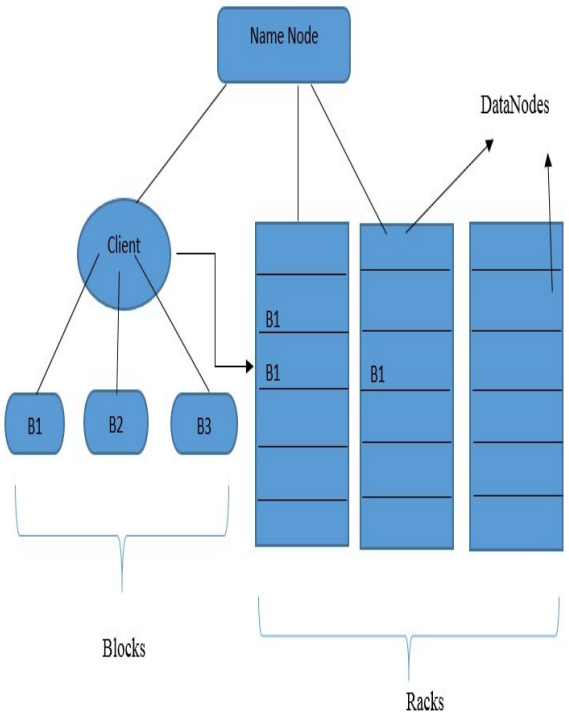


Fig. 4 Proposed replica allocation of single block on Datanodes

In the given figure it is show how the block replicas are placed in the rack. Here the client creates three replicas and places first two in any rack that was selected by the Namenode.

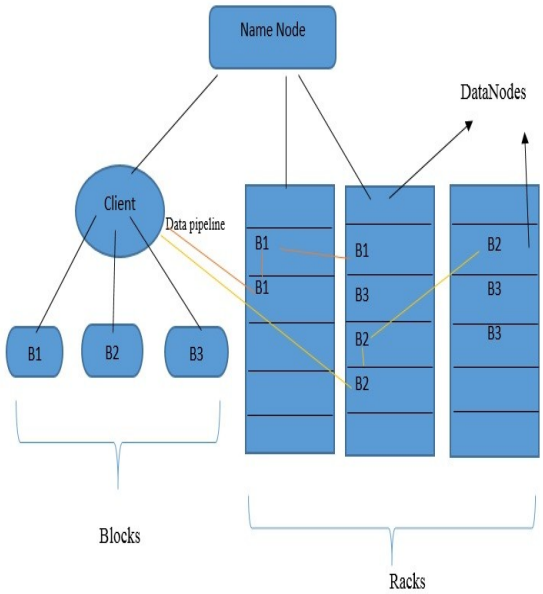


Fig. 5 Proposed replica allocation of Multi blocks on Datanodes

But it is made mandatory to place two replicas under single rack. The third replica can be placed in the rack that was next nearer to the present rack where the two block replicas are placed.Once the Replicas of first block gets placed in the Datanodes, the replicas of second block was made to get arranged in the Datanodes as like the first block. Each block Replicas have their own Data Pipelining. The acknowledgement from third replica is sent to second replica and in flow from second replica to first and reaching client from first replica.

A. Block Racking Algorithm

This algorithm will give the methodology to place the block replicas on the rack effectively and efficiently. Here in this algorithm, i represents set of entire replica and j represents the entire block set.

The range of replicas varies from i_1,i_2,i_3……. I_x. Where x represents the total number of replicas. Similarly, the range of blocks varies from j_1,j_2,j_3,….. j_y.. Where y represents the total number of blocks. Here, n in the algorithm represents the set of all available Racks and n_1,n_2,…….n_x represents the list of all individual racks in the File system.

Algorithm

1. request for the Datanode to Namenode
2. if Namenode accepts the request
3. then do
4. for every j create new i
5. for (i equals to n)
6. place i_1 i_2 on n_1
7. I_1 pipelined with i_2 of the Data;
8. for (i_x; i equals to n)
9. place i_x in the block n_x.
10. increment the value of x
11. i_x pipelines with the i_{x+1}
12. end

IV Result And Discussion

As a result of implementing the Block Racking algorithm, the way of arranging the replicas are altered. As a result, initial Datanode failure if any will make the client look for its replica very nearer that is at the same Rack. By the proposed methodology

- A rack failure makes to jump to other rack for replica.
- A Datanode failure unlike jumping to other rack as in existing methodology, jumps to next node within the rack.

For the perfect working of the block Racking Algorithm, the Namenode initially should select the first rack for the first replica of the block.

V Conclusion

The net efficiency of data write or read, manipulation, sharing, etc., gets highly increased by implementing the simple Block Racking Algorithm. The throughput time of the HDFS architecture at the time will be reduced considerably.When compared to existing methodology where a single node failure initially makes the client to change the entire rack of the existing file system.

VI Reference

[1]Big Data Processing with Hadoop-MapReduce in Cloud Systems,Rabi Prasad Padhy,Senior Software Engg, Oracle Corp.,Bangalore, Karnataka, India

[2]Critical Study of Hadoop Implementation and Performance Issues,Madhavi Vaidya,Asst. Professor, Dept of Computer Sc., Vivekanand College, Mumbai, India.

[3] http://hortonworks.com/hadoop/hdfs/

[4]http://hadoop.apache.org/docs/r1.0.4/hdfs_design.html

[5]Guanying Wang.,"Evaluating MapReduce System Performance: A Simulation Approach", August 2012

[6] Marko Grobelnik, "Big-Data Tutorial" Stavanger, May 2012.

[7]Robert D. Schneider,"Hadoop for Dummies", John Willey & sons, 2012.

Mr. B.Thirunavukarasu is presently pursuing B.E Computer Science & Engineering, SNS College of Technology, affiliated to Anna University-Chennai,Tamilnadu, India. His research interests includes BigData , Data Mining and Business Analytics. He has published a paper in National conference and in journal. He is an active entrepreneur involving web services and mobile application development.

Mrs. K.Sangeetha is presently Assistant Professor at Department of Computer Science & Engineering, SNS College of Technology, affiliated to Anna University- Chennai, Tamilnadu, India.She received the B.E degree from Sasurie College of Engineering and M.E degree from Nandha Engineering College. Currently she is pursuing her doctoral degree in AnnaUniversity Chennai. Her research interests include Datamining, and Networking. She has published many papers in international journals, national and International conference.

Dr.T.Kalaikumaran is presently Professor & HoD in the Department of Computer Science & Engineering, SNS College of Technology, affiliated to Anna University, Chennai Tamilnadu, India. He received the M.E degree from the Anna University Chennai and Ph.D degree from Ann University, Chennai.. He is interested in the research areas of data mining, spatial data mining, machine learning, uncertain data classification and clustering, pattern recognition, database management system and informational retrieval system. He is a member of CSI and IEEE

Dr.S.Karthik is presently Professor & Dean in the Department of Computer Science & Engineering, SNS College of Technology, affiliated to Anna University- Coimbatore, Tamilnadu, India. He received the M.E degree from the Anna University Chennai and Ph.D degree from Anna University of Technology, Coimbatore. His research interests include network security, web services and wireless systems. In particular, he is currently working in a research group developing new Internet security architectures and active defense systems against DDoS attacks. Dr.S.Karthik published more than 35 papers in refereed international journals and 25 papers in conferences and has been involved many international conferences as Technical Chair and tutorial presenter. He is an active member of IEEE, ISTE, IAENG, IACSIT and Indian Computer Society.

